REMARKS

In the Official Action mailed on **23 March 2006**, the Examiner reviewed claims 1-27. Claims 1-6, 10-19, and 23-27 were rejected under 35 U.S.C. §103(a) as being unpatentable over Rajwar et al (*Speculative Lock Elision*; ACM/IEEE International Symposium; Dec. 2001, hereinafter "Rajwar"), and further in view of Jim Gray (*The Transaction Concept: Virtues and Limitations*, hereinafter "Gray") and Microsoft Computer Dictionary (Fifth edition, published in 2002, hereinafter "MCD"). Claims 7-9 and 20-22 were rejected under 35 U.S.C. §103(a) as being unpatentable over Rajwar, in view of Gray, in view of MCD and further in view of Gaskins et al (USPN 6,681,311, hereinafter "Gaskins").

Rejections under 35 U.S.C. §103(a)

Independent claims 1, 14, and 27 were rejected as being unpatentable over Rajwar and further in view of Gray and MCD. Applicant respectfully points out that Rajwar teaches eliding a lock-acquire operation and speculatively executing a critical section of code (see Rajwar page 294, column 1, paragraph 2). The invention of Rajwar is "implemented entirely in microarchitecture, *without instruction set support* and *without system-level modifications*..." (see Rajwar page 295, column 2, Point 3). In order to elide a lock-acquire operation, the invention of Rajwar observes "load and store **sequences** and the values read and to be written" (see Rajwar page 297, column 2, paragraph 2). This involves looking for a **specific pattern of instructions** (see Rajwar, page 297, column 2, paragraph 2; and page 298, column 1, paragraph 1 – Point 1 of algorithm for SLE). Hence, looking at a load instruction **by itself does not** indicate whether a lock-acquire operation can be elided.

The invention of Rajwar then **predicts** that the memory operations in the critical section will occur atomically and enters a speculative execution mode to execute the critical section (see Rajwar page 298, column 1, steps 1-5). During

this speculative execution mode, memory accesses are monitored to determine whether or not an interfering memory access has occurred (see Rajwar page 298, section 3.4).

In contrast, the present invention elides a lock by *explicitly executing* a "start transaction execution" (STE) instruction before entering a critical section (see paragraph [0010] and [0059] of the instant application). Since a transaction is explicitly entered, indicating that the critical section is to be executed atomically, no prediction is required.

During transactional execution, the present invention monitors load-marked cache lines to determine if an interfering memory access has occurred (see paragraphs [0011] and [0062] of the instant application). A cache line becomes load-marked when the present invention encounters a load instruction that indicates that the cache line should be load-marked (see paragraphs [0088]-[0089] of the instant application).

Prior to load-marking a cache line, the present invention determines whether the load instruction is a monitored load instruction or an unmonitored load instruction by **analyzing the load instruction** (see paragraphs [0082]-[0084], and [0086] of the instant application). If the load instruction is a monitored load instruction, the present invention load-marks the cache line (see paragraph [0089] of the instant application).

Note that analysis of the load instruction can involve: (1) determining whether the load operation is directed to a heap, (2) examining a data structure associated with the load operation to determine if the data structure is a protected data structure for which loads need to be monitored, (3) examining the op code for the load instruction, and (4) examining an address associated with the load instruction to determine whether the address falls within a range of addresses for which loads are monitored (see paragraphs [0082]-[0084], and [0086] of the instant application). In all cases, the load instruction **itself** is monitored. **No**

other instructions are monitored to determine if the load instruction is a monitored load instruction.

There is nothing within Rajwar or Gray, either separately or in concert, which suggests eliding a lock by (1) *explicitly executing a start transaction execution instruction*, (2) encountering a load instruction during transactional execution, (3) determining whether the load instruction is a monitored load instruction or an unmonitored load instruction by *analyzing the load instruction*, and (4) if the load instruction is a monitored load instruction, performing the load operation and load-marking a cache line associated with the load instruction.

Accordingly, Applicant has amended independent claims 1, 14, and 27 to clarify that the present invention determines whether the load instruction is a monitored load instruction or an unmonitored load instruction by **analyzing the load instruction.** These amendments find support in paragraphs [0082]-[0084], and [0086] of the instant application.

Applicant has also amended claims 6 and 7 to insert the word "wherein." No new matter has been added to claims 6 and 7.

Hence, Applicant respectfully submits that independent claims 1, 14, and 27 as presently amended are in condition for allowance. Applicant also submits that claims 2-13, which depend upon claim 1, and claims 15-26, which depend upon claim 14, are for the same reasons in condition for allowance and for reasons of the unique combinations recited in such claims.

CONCLUSION

It is submitted that the present application is presently in form for allowance. Such action is respectfully requested.

Respectfully submitted,

By _____
Edward J. Grundler
Registration No. 47,615

Date: 27 April 2006

Edward J. Grundler
PARK, VAUGHAN & FLEMING LLP
2820 Fifth Street
Davis, CA 95616-7759
Tel: (530) 759-1663
FAX: (530) 759-1665
Email: edward@parklegal.com